PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2001-306581
(43)Date of publication of application : 02.11.2001

(51)Int.Cl. G06F 17/30

(21)Application number : 2000-116392
(22)Date of filing : 18.04.2000

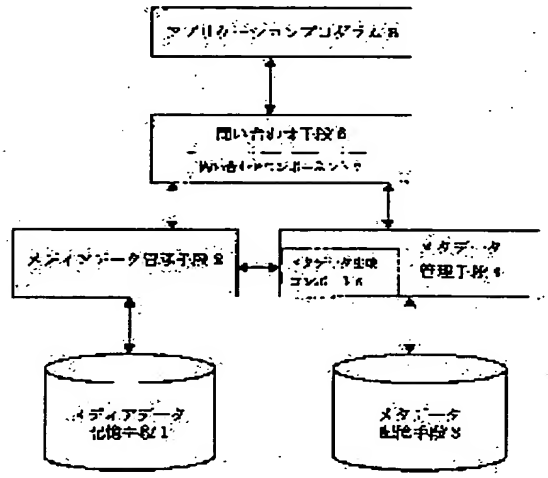
(71)Applicant : SONY CORP
(72)Inventor : ASAZU HIDEKI

(54) MIDDLEWARE AND MEDIA DATA AUDIOVISUAL EQUIPMENT USING THE MIDDLEWARE

(57)Abstract:

PROBLEM TO BE SOLVED: To provide a middleware, capable of executing semantic access or synthesizing processing to the prescribed part of media data by using meta-data, and efficiently developing an application having a function therefor.

SOLUTION: In the middleware to be operated in the media data audiovisual equipment, this middleware has a media data managing means 2 for providing a managing function which includes reproducing, read, recording, deletion and synthesizing of media data and recording of the history of access, a meta-data managing means 4 for providing a managing function inducing read, recording and dynamic generation of meta-data and transaction processing, and an inquiry means 6, having a function for providing the interface of higher abstraction degree to an application program 8 by substituting processing for access to the media data managing means and the meta-data managing means.



LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号
特開2001－306581
（Ｐ2001－306581Ａ）
(43)公開日　平成13年11月2日(2001.11.2)

(51)Int.Cl.'	識別記号	ＦＩ	ﾃｰﾏｺｰﾄﾞ(参考)
Ｇ06Ｆ　17/30	170	Ｇ06Ｆ　17/30	170Ｇ　5Ｂ075

審査請求　未請求　請求項の数10　ＯＬ　（全 25 頁）

(21)出願番号　特願2000－116392(Ｐ2000－116392)

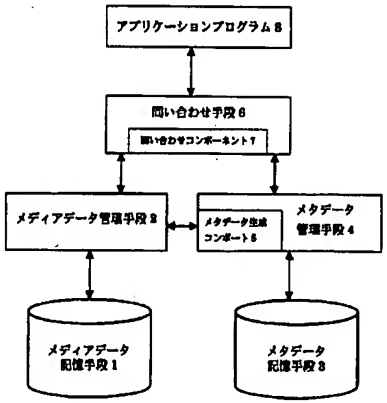
(22)出願日　平成12年４月18日(2000.4.18)

(71)出願人　000002185
ソニー株式会社
東京都品川区北品川６丁目７番35号
(72)発明者　浅沖　英樹
東京都品川区北品川６丁目７番35号　ソニー株式会社内
(74)代理人　100078031
弁理士　大石　皓一　（外２名）

Ｆターム(参考)　5Ｂ075 ND16 NS10 PQ02 PQ46 PR03
UU40

(54)【発明の名称】　ミドルウェアおよびミドルウェアを用いたメディアデータ視聴機器

(57)【要約】　（修正有）
【課題】　メタデータを用いることによって、メディアデータに対して、所定部分の意味的なアクセスや合成処理を実行することができ、これらの機能を有したアプリケーションを効率的に開発することができるミドルウェアを提供する。
【解決手段】　メディアデータ視聴機器において動作するミドルウェアであって、メディアデータの再生、読み出し、記録、削除および合成ならびにアクセスの履歴の記録を含む管理機能を提供するメディアデータ管理手段２と、メタデータの読み出し、記録および動的生成ならびにトランザクション処理を含む管理機能を提供するメタデータ管理手段４と、メディアデータ管理手段およびメタデータ管理手段へのアクセス処理を代行し、アプリケーションプログラム８に対して、より抽象度の高いインターフェイスを提供する機能を有する問い合わせ手段６を有することを特徴とするミドルウェア。



【特許請求の範囲】
【請求項１】　メディアデータを蓄積し、再生する機能を有するメディアデータ視聴機器において動作するミドルウェアであって、メディアデータの再生、読み出し、記録、削除および合成ならびにメディアデータに対するアクセスの履歴の記録を含むメディアデータの管理機能を提供するメディアデータ管理手段と、メタデータの読み出し、記録および動的生成ならびにメタデータ対するアクセスにともなうトランザクション処理を含むメタデータの管理機能を提供するメタデータ管理手段と、前記メディアデータ管理手段および前記メタデータ管理手段へのアクセス処理を代行し、アプリケーションプログラムに対して、より抽象度の高いアクセスインターフェイスを提供する機能を有する問い合わせ手段を有し、全体として、前記メディアデータのデータベースとしての機能を、前記アプリケーションプログラムに対して提供することを特徴とするミドルウェア。
【請求項２】　前記メディアデータ管理手段において、前記メディアデータに対するアクセスの履歴を記録する際に、アクセス履歴情報をメタデータに変換し、前記メタデータ管理手段に記録することを特徴とする請求項１に記載のミドルウェア。
【請求項３】　前記メディアデータ管理手段において、前記メディアデータの記録をおこなう際に、入力されたメディアデータについて、あらかじめ登録されたデータ解析処理を適用することによって、メタデータを作成し、前記メタデータ管理手段に記録することを特徴とする請求項１または２に記載のミドルウェア。
【請求項４】　前記メタデータ管理手段が有するメタデータを動的に生成する機能を、独立したプラグ・イン・コンポーネントとして分離し、前記プラグ・イン・コンポーネントをシステムに動的に組み込む機能、使用しなくなった前記プラグ・イン・コンポーネントを破棄する機能およびすでに組み込まれているプラグ・イン・コンポーネントのうち、アプリケーションプログラムの目的に合致したプラグ・イン・コンポーネントを自動的に選択し、実行する機能を提供することを特徴とする請求項１ないし３のいずれか１項に記載のミドルウェア。
【請求項５】　前記問い合わせ手段が有する前記メディアデータ管理手段および前記メタデータ管理手段へのアクセス処理を代行し、アプリケーションプログラムに対して、より抽象度の高いアクセスインターフェイスを提供する機能を独立したプラグ・イン・コンポーネントとして分離し、前記プラグ・イン・コンポーネントをシステムに動的に組み込む機能、使用しなくなった前記プラグ・イン・コンポーネントを破棄する機能およびすでに組み込まれているプラグ・イン・コンポーネントのうち、アプリケーションプログラムから提示された条件に合致したプラグ・イン・コンポーネントを検索する機能を提供することを特徴とする請求項１ないし４のいずれか１項に記載のミドルウェア。
【請求項６】　メディアデータを記録し、再生する機能を有するメディアデータ視聴機器であって、前記メディアデータを格納するメディアデータ記録デバイスと、メタデータを格納するメタデータ記録デバイスと、アプリケーションプログラムを解釈実行するプロセッサとを有し、さらに、メディアデータの再生、読み出し、記録、削除および合成ならびにメディアデータに対するアクセスの履歴の記録を含むメディアデータの管理機能を提供するメディアデータ管理手段と、メタデータの読み出し、記録および動的生成ならびにメタデータ対するアクセスにともなうトランザクション処理を含むメタデータの管理機能を提供するメタデータ管理手段と、前記メディアデータ管理手段および前記メタデータ管理手段へのアクセス処理を代行し、アプリケーションプログラムに対して、より抽象度の高いアクセスインターフェイスを提供する機能を有する問い合わせ手段を有し、全体として、前記メディアデータのデータベースとしての機能を、前記アプリケーションプログラムに対して提供するミドルウェアを搭載したことを特徴とするメディアデータ視聴機器。
【請求項７】　前記メディアデータ管理手段において、前記メディアデータに対するアクセスの履歴を記録する際に、アクセス履歴情報をメタデータに変換し、前記メタデータ管理手段に記録することを特徴とする請求項６に記載のメディアデータ視聴機器。
【請求項８】　前記メディアデータ管理手段において、前記メディアデータの記録をおこなう際に、入力されたメディアデータについて、あらかじめ登録されたデータ解析処理を適用することによって、メタデータを作成し、前記メタデータ管理手段に記録することを特徴とする請求項６または７に記載のメディアデータ視聴機器。
【請求項９】　前記メタデータ管理手段が有するメタデータを動的に生成する機能を、独立したプラグ・イン・コンポーネントとして分離し、前記プラグ・イン・コンポーネントをシステムに動的に組み込む機能、使用しなくなった前記プラグ・イン・コンポーネントを破棄する機能およびすでに組み込まれているプラグ・イン・コンポーネントのうち、アプリケーションプログラムの目的に合致したプラグ・イン・コンポーネントを自動的に選択し、実行する機能を提供することを特徴とする請求項６ないし８のいずれか１項に記載のメディアデータ視聴機器。
【請求項１０】　前記問い合わせ手段が有する前記メディアデータ管理手段および前記メタデータ管理手段へのアクセス処理を代行し、アプリケーションプログラムに対して、より抽象度の高いアクセスインターフェイスを提供する機能を独立したプラグ・イン・コンポーネントとして分離し、前記プラグ・イン・コンポーネントをシステムに動的に組み込む機能、使用しなくなった前記プ

3

ラグ・イン・コンポーネントを破棄する機能およびすでに組み込まれているプラグ・イン・コンポーネントのうち、アプリケーションプログラムから提示された条件に合致したプラグ・イン・コンポーネントを検索する機能を提供することを特徴とする請求項6ないし9のいずれか1項に記載のメディアデータ視聴機器。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、ミドルウェアおよびミドルウェアを用いたメディアデータ視聴機器関するものであり、さらに詳細には、メタデータを用いることによって、メディアデータに対して、ある俳優の映っている部分などの意味的なアクセスをおこなうことを可能とするとともに、メディアデータのダイジェストの生成などのメディアデータの合成処理を実行することができ、かつ、これらの機能を有したアプリケーションプログラムを効率的に開発することができるミドルウェアおよびそれを用いたメディアデータ視聴機器に関するものである。

【0002】

【従来の技術】ディジタル技術の発達によって、メディアデータを劣化させることなく、大量に蓄積し、利用することが可能になって来ている。とくに、近年、プロセッサや記録デバイスの低価格化にともなって、記録デバイスにハードディスクを用いた民生向けのディジタル記録機器が提供されるようになっている。

【0003】これらのディジタル記録機器においては、ハードウエアの機能のみならず、ハードウエアの機能を活かして、何を可能にするかが重要であり、ソフトウエアがより重要なものとして、認識されるようになっている。そこで、魅力的なアプリケーションプログラムを効率的に開発することが必要とされている。

【0004】しかしながら、従来は、各機器毎に、個別に、アプリケーションプログラムを開発しており、アプリケーションプログラムの開発には高いコストが必要であった。

【0005】とりわけ、大量のメディアデータから、アプリケーションプログラムにとって必要な部分を取り出すという処理は、機器の性格上、多くのアプリケーションプログラムが共通して、必要とする処理であるが、従来は、ＥｌｅｃｔｒｏｎｉｃＰｒｏｇｒａｍｍｉｎｇ Ｇｕｉｄｅ（ＥＰＧ）にアクセスする方法しか利用ができなかった。

【0006】この方法は、ＥＰＧを利用することにより、ジャンルや出演俳優などの情報をキーとして、コンテンツデータ単位で、メディアデータにアクセスすることが可能になるものの、各メディアデータの内容に、所望のように、アクセスすることには限界があった。

【0007】すなわち、たとえば、映像データの場合、映像データは、通常、複数のフレーム画像が時間的に連

4

続して、並んだものとして扱われており、この方法でも、ＥＰＧを利用することにより、時刻やフレーム番号を指定して、フレーム画像にアクセスすることができる。

【0008】しかしながら、ＥＰＧを利用する方法では、ある俳優の映っている部分のみを映像データから抽出し、連続して再生することや、特定のシーンのみを繋ぎ合わせて、ダイジェストを作ることなどは困難であり、各メディアデータの内容に、所望のように、アクセスすることには限界があった。

【0009】

【発明が解決しようとする課題】各メディアデータの内容に、所望のように、アクセスするための一つの方法として、メタデータを用いる方法が挙げられる。ここに、メタデータとは、ｄａｔａ ａｂｏｕｔ ｄａｔａを意味し、メディアデータへのアクセスを補助するためのデーター般を指し、ＥＰＧもこれに含まれる。

【0010】メタデータには、「映像、音声などがどのようなシーンに区切られるか」や、「各シーンやフレームに、何あるいは誰が映っているか」などの情報や、「個々のユーザーがこれまでに、どのコンテンツデータあるいはシーンを視聴したか」や、「各コンテンツデータあるいはシーンが、何時、誰によって視聴されたか」などの情報も含まれており、したがって、メタデータを利用することによって、「ある俳優が、どのメディアデータのどの部分に映っているか」などを知ることができ、これによって、「ある俳優の映っているシーンを取り出す」など、メディアデータに対して、その内容に基づいたアクセスをすることが可能になる。

【0011】しかしながら、メタデータは、メディアデータのフォーマットの中に埋め込まれている場合、データ放送によって、メディアデータとともに配信される場合、インターネットなど、外部のネットワークを通じて、取得する場合、機器側で、メディアデータについての解析処理をおこない、自動的に生成する場合、ユーザーの視聴履歴などのように、メディアデータの視聴にともなって、生成される場合、ユーザー自身によって生成される場合など、種々のソースから、種々の形式で取得されるため、これらのメタデータを利用して、メディアデータにアクセスするアプリケーションプログラムを作成することは容易なことではなく、メタデータを利用して、メディアデータにアクセスするアプリケーションプログラムを開発するには大きな労力が必要であった。

【0012】したがって、本発明は、メタデータを用いることによって、メディアデータに対して、ある俳優の映っている部分などの意味的なアクセスをおこなうことを可能とするとともに、メディアデータのダイジェストの生成などのメディアデータの合成処理を実行することができ、かつ、これらの機能を有したアプリケーションプログラムを効率的に開発することができるミドルウェアおよびそれを用いたメディアデータ視聴機器を提供することを目的とするものである。

5

【0013】

【課題を解決するための手段】本発明のかかる目的は、メディアデータを蓄積し、再生する機能を有するメディアデータ視聴機器において動作するミドルウェアであって、メディアデータの再生、読み出し、記録、削除および合成ならびにメディアデータに対するアクセスの履歴の記録を含むメディアデータの管理機能を提供するメディアデータ管理手段と、メタデータの読み出し、記録および動的生成ならびにメタデータ対するアクセスにともなうトランザクション処理を含むメタデータの管理機能を提供するメタデータ管理手段と、前記メディアデータ管理手段および前記メタデータ管理手段へのアクセス処理を代行し、アプリケーションプログラムに対して、より抽象度の高いアクセスインターフェイスを提供する機能を有する問い合わせ手段を有し、全体として、メディアデータのデータベースとしての機能を、前記アプリケーションプログラムに対して提供することを特徴とするミドルウェアによって達成される。

【0014】本発明によれば、メタデータ管理手段によって管理されるメタデータを用いることによって、メディアデータに対して、ある俳優の映っている部分などの意味的なアクセスをおこなうことが可能になるとともに、メディアデータ管理手段の機能によって、メディアデータのダイジェストの生成などのメディアデータの合成処理を実行することが可能になり、かつ、問い合わせ手段の機能によって、これらの機能を有したアプリケーションプログラムを効率的に開発することができる。

【0015】本発明の好ましい実施態様においては、前記メディアデータ管理手段において、前記メディアデータに対するアクセスの履歴を記録する際に、アクセス履歴情報をメタデータに変換し、前記メタデータ管理手段に記録するように構成されている。

【0016】本発明の好ましい実施態様によれば、ユーザーの視聴履歴情報についても、他のメタデータと同様に、ミドルウェアおよびアプリケーションプログラムから統一的に扱うことが可能になるから、アプリケーションプログラムのサイズを小さく保つことができ、また、アクセス履歴情報をメタデータに変換することによって、同じミドルウェアを搭載している他の視聴機器の間での視聴履歴情報の交換が可能になる。

【0017】本発明のさらに好ましい実施態様においては、前記メディアデータ管理手段において、前記メディアデータの記録をおこなう際に、入力されたメディアデータについて、あらかじめ登録されたデータ解析処理を適用することによって、メタデータを作成し、前記メタデータ管理手段に記録するように構成されている。

【0018】本発明のさらに好ましい実施態様によれば、メディアデータとともに、メタデータが提供されな

6

い場合でも、メディアデータ視聴機器側で、メタデータを生成することが可能となり、メタデータを利用したアプリケーションプログラムを動作させることが可能になる。

【0019】本発明のさらに好ましい実施態様においては、前記メタデータ管理手段が有するメタデータを動的に生成する機能を、独立したプラグ・イン・コンポーネントとして分離し、前記プラグ・イン・コンポーネントをシステムに動的に組み込む機能、使用しなくなった前記プラグ・イン・コンポーネントを破棄する機能およびすでに組み込まれているプラグ・イン・コンポーネントのうち、アプリケーションプログラムの目的に合致したプラグ・イン・コンポーネントを自動的に選択し、実行する機能を提供するように構成されている。

【0020】本発明のさらに好ましい実施態様によれば、メタデータを動的に組み込むことが可能になり、種々のソースから得られるメタデータをミドルウェアが解釈できる形に変換して、統一的に扱うことが可能となるとともに、将来、新しいアルゴリズム／フォーマットが開発された場合でも、ミドルウェア／システムを変更することなしに、それらの新しいアルゴリズム／フォーマットを導入することができる。

【0021】本発明のさらに好ましい実施態様においては、前記問い合わせ手段が有する前記メディアデータ管理手段および前記メタデータ管理手段へのアクセス処理を代行し、アプリケーションプログラムに対して、より抽象度の高いアクセスインターフェイスを提供する機能を独立したプラグ・イン・コンポーネントとして分離し、前記プラグ・イン・コンポーネントをシステムに動的に組み込む機能、使用しなくなった前記プラグ・イン・コンポーネントを破棄する機能およびすでに組み込まれているプラグ・イン・コンポーネントのうち、アプリケーションプログラムから提示された条件に合致したプラグ・イン・コンポーネントを検索する機能を提供するように構成されている。

【0022】本発明のさらに好ましい実施態様によれば、ミドルウェアの機能を必要に応じて拡張することが可能となり、アプリケーションプログラムの多彩な要求に応えることができ、また、アプリケーションプログラムの開発を、メタデータ管理手段やメディアデータ管理手段の機能を用いて、基本的な機能を開発する開発者と、それらの機能を利用して、アプリケーションプログラム全体の機能を開発する開発者とで分業することが可能となって、アプリケーションプログラムの開発効率をさらに向上させることができる。

【0023】本発明の前記目的はまた、メディアデータを記録し、再生する機能を有するメディアデータ視聴機器であって、前記メディアデータを格納するメディアデータ記録デバイスと、メタデータを格納するメタデータ記録デバイスと、アプリケーションプログラムを解釈実

行するプロセッサとを有し、さらに、メディアデータの再生、読み出し、記録、削除および合成ならびにメディアデータに対するアクセスの履歴の記録を含むメディアデータの管理機能を提供するメディアデータ管理手段と、メタデータの読み出し、記録および動的生成ならびにメタデータ対するアクセスにともなうトランザクション処理を含むメタデータの管理機能を提供するメタデータ管理手段と、前記メディアデータ管理手段および前記メタデータ管理手段へのアクセス処理を代行し、アプリケーションプログラムに対して、より抽象度の高いアクセスインターフェイスを提供する機能を有する問い合わせ手段を有し、全体として、前記メディアデータのデータベースとしての機能を、前記アプリケーションプログラムに対して提供するミドルウェアを搭載したことを特徴とするメディアデータ視聴機器によって達成される。

【0024】本発明によれば、メディアデータ視聴機器が有するメディアデータを蓄積するという特徴を活かして、アプリケーションプログラムを動作させることが可能となり、膨大な量のメディアデータを有効活用することが可能になる。

【0025】本発明の好ましい実施態様においては、メディアデータ視聴機器の前記メディアデータ管理手段において、前記メディアデータに対するアクセスの履歴を記録する際に、アクセス履歴情報をメタデータに変換し、前記メタデータ管理手段に記録するように構成されている。

【0026】本発明のさらに好ましい実施態様においては、メディアデータ視聴機器の前記メディアデータ管理手段において、前記メディアデータの記録をおこなう際に、入力されたメディアデータについて、あらかじめ登録されたデータ解析処理を適用することによって、メタデータを作成し、前記メタデータ管理手段に記録するように構成されている。

【0027】本発明のさらに好ましい実施態様においては、メディアデータ視聴機器の前記メタデータ管理手段が有するメタデータを動的に生成する機能を、独立したプラグ・イン・コンポーネントとして分離し、前記プラグ・イン・コンポーネントをシステムに動的に組み込む機能、使用しなくなった前記プラグ・イン・コンポーネントを破棄する機能およびすでに組み込まれているプラグ・イン・コンポーネントのうち、アプリケーションプログラムの目的に合致したプラグ・イン・コンポーネントを自動的に選択し、実行する機能を提供するように構成されている。

【0028】本発明のさらに好ましい実施態様においては、メディアデータ視聴機器の前記問い合わせ手段が有する前記メディアデータ管理手段および前記メタデータ管理手段へのアクセス処理を代行し、アプリケーションプログラムに対して、より抽象度の高いアクセスインターフェイスを提供する機能を独立したプラグ・イン・コンポーネントとして分離し、前記プラグ・イン・コンポーネントをシステムに動的に組み込む機能、使用しなくなった前記プラグ・イン・コンポーネントを破棄する機能およびすでに組み込まれているプラグ・イン・コンポーネントのうち、アプリケーションプログラムから提示された条件に合致したプラグ・イン・コンポーネントを検索する機能を提供するように構成されている。

【0029】

【発明の実施の形態】以下、添付図面に基づいて、本発明の好ましい実施態様につき、詳細に説明を加える。

【0030】図1は、本発明の実施態様にかかるミドルウエアを含むメディアデータ視聴機のブロックダイアグラムである。

【0031】図1に示されるように、メディアデータ視聴機は、メディアデータを記憶するメディアデータ記憶手段1と、メディアデータ記憶手段1に記憶されたメディアデータの再生、読み出しおよび合成、メディアデータ記憶手段1へのメディアデータの記録ならびにメディアデータに対するアクセスの履歴の記録など、メディアデータの管理をおこなうメディアデータ管理手段2と、メタデータを記憶するメタデータ記憶手段3と、メタデータ記憶手段3に記憶されたメタデータ読み出し、メタデータ記憶手段3へのメタデータの記録、メタデータの動的生成およびアクセスにともなうトランザクション処理など、メタデータの管理をおこなうメタデータ管理手段4と、メディアデータを解析して、メタデータを動的に生成し、ネットワークなどを通じて、システム外部から提供されるメタデータを、メタデータ管理手段4が扱うことのできる形式のメタデータに変換するなどの処理を実行するメタデータ生成コンポーネント5と、メディアデータ管理手段2およびメタデータ管理手段4へのアクセス処理を隠蔽し、プログラミングに習熟していないユーザーが処理可能なより平易なアクセスインターフェイスをアプリケーションプログラム8に対して提供するモジュールである問い合わせ手段6と、問い合わせ手段6から、プラグインコンポーネントととして分離した独立のアクセスインターフェイスからなる問い合わせコンポーネント7を備えている。

【0032】本実施態様において、メタデータ生成コンポーネント5は、メタデータ管理手段4から分離されており、これによって、種々のデータ解析アルゴリズム、メタデータのローダを、システムに動的に追加することが可能となり、種々のソースから取得されるメタデータを統一的に扱うことが可能になる。

【0033】また、本実施態様においては、プログラミングに習熟していないユーザーが処理可能なより平易なアクセスインターフェイスをアプリケーションプログラム8に対して提供するモジュールである問い合わせ手段6を設けているので、アプリケーションプログラム8側の負担を軽減することが可能になる。すなわち、たとえば、ある俳優の映っている部分のみを抽出し、連続して再生するという処理は、メディアデータ管理手段2およびメタデータ管理手段4に対するいくつかの処理に分割することができるが、これらの処理を、その都度、アプリケーションプログラム8側でおこなうことは、きわめて煩雑である。そこで、本実施態様においては、問い合わせ手段6において、より抽象度が高く、プログラミングに習熟していないユーザーでも処理可能なアクセスインターフェイスを提供し、アプリケーションプログラム8側の負担を軽減している。

【0034】さらに、本実施態様においては、アクセスインターフェイスを独立したプラグインコンポーネントととして分離して、問い合わせコンポーネント7を構成しているので、これらの処理を問い合わせ手段6から分離し、システムに対して、動的に組み込むことができ、問い合わせ手段6を後に容易に拡張することが可能となり、種々のアプリケーションプログラムからの様々な要求に応えることができる。

【0035】以下、サン・マイクロシステムズ株式会社によって開発されたＪａｖａ（登録商標）を用いた場合につき、本実施態様を詳細に説明する。

【0036】図2は、システムを構成するソフトウエアの階層図である。

【0037】図2に示されるように、本発明の実施態様にかかるミドルウエアを含むメディアデータ視聴機上においては、ハードウエア資源の管理をおこなうオペレーティングシステム上で、Ｊａｖａのバイトコードの解釈実行をおこなうＪａｖａＶＭ（Ｊａｖａ仮想機械）が動作する。さらに、Ｊａｖａの実行環境を構成する基本アプリケーションプログラムインターフェイスおよびその他の拡張アプリケーションプログラムインターフェイスが提供されている。

【0038】アプリケーションプログラムは、これらのＪａｖａアプリケーションプログラムを利用して、記述あるいは実行される。本実施態様にかかるミドルウエアは、Ｊａｖａの拡張アプリケーションプログラムインターフェイスとして提供され、アプリケーションプログラムに対し、メディアデータおよびメタデータを取り扱うための機能を提供している。

【0039】図3は、本実施態様にかかるミドルウエアのパッケージ構成を示すパッケージ図であり、UML（Ｕｎｉｆｉｅｄ Ｍｏｄｅｌｉｎｇ Ｌａｎｇｕａｇｅ）を用いたものである。

【0040】図3に示されるように、本実施態様にかかるミドルウエアは、4つのパッケージと2種類のプラグ・イン・コンポーネントによって構成されている。

【0041】図3に示されるように、ミドルウエアは、図1におけるメディアデータ管理手段2の機能を実現するメディアデータ・データベース・パッケージ10と、図1におけるメタデータ管理手段4の機能を実現するメタデータ・データベース・パッケージ11と、図1における問い合わせ手段6の機能を実現するクウェリー・インターフェイス・パッケージ12と、他のパッケージあるいはコンポーネントにより共通して利用されるユーティリテイ・クラスを集めたユーティリテイ・パッケージ13と、図1における問い合わせコンポーネント7の機能を実現するＱＴプラグ・イン・コンポーネント14と、図1におけるメタデータ生成コンポーネント5の機能を実現するＤＡプラグ・イン・コンポーネント15とにより構成されている。

【0042】図4は、メディアデータ・データベース・パッケージ10の詳細を示すクラスダイアグラムである。

【0043】メディアデータ・データベース・パッケージ10は、メディアデータの管理をおこない、メディアデータ記憶手段1に記憶されたメディアデータの再生、読み出し、合成、削除、メディアデータ記憶手段1へのメディアデータの記録ならびにメディアデータに対するアクセスの履歴の記録などの機能を提供するものであり、本実施態様においては、メディアデータの再生、読み出し、合成、削除、記録などの基本的な機能は、ＪａｖａのＡＰＩであるＪａｖａ ＭｅｄｉａＦｒａｍｅｗｏｒｋ（ＪＭＦ）を用いて、実現している。

【0044】図4において、Ｊａｖａｘ．ｍｅｄｉａ．Ｐｒｏｃｅｓｓｏｒ インターフェイス20は、ＪＭＦによって規定され、動画、音声などのストリームメディアデータに対し、Ｊａｖａからアクセスするためのインターフェイスを定義するものである。ＪＭＦにおいて定義されているＪａｖａｘ．ｍｅｄｉａ．Ｐｒｏｃｅｓｓｏｒ インターフェイス20においては、データのデコード、レンダリングなどの機能を、プラグ・イン・コンポーネントとして、追加することができるように構成されており、この機能を利用して、動画、音声などのストリームメディアデータにグラフィックを合成するなどの処理をおこなうことが可能になる。Ｐｒｏｃｅｓｓｏｒ 実装クラス21は、Ｊａｖａｘ．ｍｅｄｉａ．Ｐｒｏｃｅｓｓｏｒの実装クラスで、各プラットフォーム毎に提供されるものである。また、プロセッサ22は、Ｊａｖａｘ．ｍｅｄｉａ．Ｐｒｏｃｅｓｓｏｒの実装クラスで、Ｐｒｏｃｅｓｓｏｒ 実装クラス21の機能を拡張し、プレイリスト・エントリー23で指定する複数のメディアデータを合成して、一つの仮想メディアデータとして、アクセスする機能およびメディアデータへのアクセス情報を後述するロガー24へ通知する機能を提供するものであり、レコーダ25は、ジャバックス・メディア・プロセッサの実装クラスで、プロセッサ22と同様に、Ｐｒｏｃｅｓｓｏｒ 実装クラス21の機能を拡張したものであるが、とくに、メディアデータの記録に特化した機能を提供しており、メディアデータの記録時に、不慮のエラーが発生した場合でも、二次記録装置に

11

記録されたデータに不整合が生じないようにするためのトランザクション機能およびメディアデータの記録時に、レコーディング・マネージャー２６に登録されているＤＡプラグ・イン・コンポーネント１５を起動して、データ解析処理を実行し、メタデータの生成をおこなう機能を提供している。さらに、メディア・マネージャー２７は、プロセッサ２２とレコーダ２５についてのインスタンスの生成、メディアデータの削除などの機能を提供し、ロガー２４は、プロセッサ２２とレコーダ２５に対するアクセスを記録し、メディアデータ毎、あるいは、ユーザー毎の視聴履歴情報を作成する機能を有している。ロガー２４によって作成された視聴履歴情報はメタデータに変換され、メタデータ・データベース・パッケージ１１の機能を利用して、メタデータとして、メタデータ記憶手段３に記憶される。

【００４５】図５は、メディアデータを再生する手順を示すシーケンス図である。

【００４６】本実施態様においては、アプリケーションプログラム８からの問い合わせは、ＱＴプラグ・イン・コンポーネント１４によって解釈され、メディアデータ・データベース・パッケージ１０への処理要求に変換される。

【００４７】図５においては、アプリケーションプログラム８から、クエリー＃１、クエリー＃２およびクエリー＃３の３つの問い合わせが、ＱＴプラグ・イン・コンポーネント１４に対してなされ、それらに対応して、それぞれ、「プロセッサの生成」、「再生開始」、「再生停止」という処理が、メディアデータ・データベース・パッケージ１０において、実行される。

【００４８】クエリー＃１に対しては、以下の処理が実行される。

【００４９】まず、ＱＴプラグ・イン・コンポーネント１４より、メディア・マネージャー２７に対し、ｃｒｅａｔｅ Ｐｒｏｃｅｓｓｏｒ () 関数を呼び出し、引数として指定したメディアデータを再生するためのプロセッサ２２を生成させる。

【００５０】メディアデータの再生時に、メディアデータとグラフィックの合成などをおこなうときは、ＱＴプラグ・イン・コンポーネント１４は、さらに、適切なＪａｖａｘ．ｍｅｄｉａ．ＰｌｕｇＩｎを生成する。

【００５１】生成したＪａｖａｘ．ｍｅｄｉａ．ＰｌｕｇＩｎをプロセッサ２２に組み込み、メディアデータを再生する準備が完了する。

【００５２】クエリー＃２に対しては、以下の処理が実行される。

【００５３】まず、ＱＴプラグ・イン・コンポーネント１４は、プロセッサ２２に対し、ｓｔａｒｔ () 関数を呼び出し、メディアデータの再生の開始を指示する。

【００５４】次いで、プロセッサ２２は、ロガー２４に対して、メディアデータの再生が開始されることを通知する。ロガー２４は、メディアデータの再生が開始される旨の通知を受けると、メディアデータに対するアクセス履歴情報を更新し、必要があれば、新たに、メディアデータを生成して、メディアデータ・データベース・パッケージ１０を通じて、メタデータの記録をおこなう。

【００５５】次いで、メディアデータの再生が開始される。その際、必要があれば、先に組み込まれたＪａｖａｘ．ｍｅｄｉａ．ＰｌｕｇＩｎを、各フレームについて、順次、呼び出し、処理を実行する。

【００５６】クエリー＃３に対しては、以下の処理が実行される。

【００５７】ＱＴプラグ・イン・コンポーネント１４は、プロセッサ２２に対し、Ｓｔｏｐ () 関数を呼び出し、メディアデータの再生の停止を指示する。

【００５８】プロセッサ２２は、ロガー２４に対し、メディアデータの再生を停止することを通知する。ロガー２４は、メディアデータの再生を停止する旨の通知を受けると、メディアデータに対するアクセス履歴情報を更新し、必要があれば、新たに、メディアデータを生成して、メディアデータ・データベース・パッケージ１０を通じて、メタデータの記録をおこなう。

【００５９】図６は、メディアデータを記録する際の手順を示すシーケンス図である。

【００６０】本実施態様においては、すべての処理に先立って、システムは、まず、メディアデータを記録する際に、自動的に起動すべきＤＡプラグ・イン・コンポーネント１５を、レコーディング・マネージャー２６に対し、登録する。登録されたＤＡプラグ・イン・コンポーネント１５は、メディアデータの解析処理を実行し、自動的にメタデータを生成する。たとえば、映像のシーンの切れ目を検出し、メタデータとして記録する、映像中に映っている人物を特定し、メタデータとして記録する、映像中に含まれているキャプションを検出し、メタデータとして記録する、音声情報に対して、話者認識を実行し、メタデータとして記録するなどの処理がなされる。

【００６１】次いで、メディアデータの再生の場合と同様に、アプリケーションプログラム８から、クエリー＃１、クエリー＃２およびクエリー＃３の３つの問い合わせが、ＱＴプラグ・イン・コンポーネント１４に対してなされ、それらに対応して、それぞれ、「レコーダの生成」、「記録開始」、「記録停止」という処理が、メディアデータ・データベース・パッケージ１０において、実行される。

【００６２】クエリー＃１に対しては、以下の処理が実行される。

【００６３】まず、ＱＴプラグ・イン・コンポーネント１４は、メディア・マネージャー・２７に対し、ｃｒｅａｔｅ Ｒｅｃｏｒｄｅｒ () 関数を呼び出して、レコーダ・クラス２５を生成するように指示する。

13

【００６４】メディア・マネージャー・クラス２７は、レコーダ・クラス２５のオブジェクトを生成して、ＱＴプラグ・イン・コンポーネント１４に返す。

【００６５】クエリー＃２に対しては、以下の処理が実行される。

【００６６】まず、ＱＴプラグ・イン・コンポーネント１４は、レコーダ２５に対して、ｓｔａｒｔ () 関数を呼び出し、メディアデータの記録の開始を指示する。

【００６７】次いで、レコーダ２５は、ロガー２４に対し、メディアデータの記録が開始されることを通知する。

【００６８】その後、レコーダ２５は、メディアデータの記録を実行する。その際、各フレームを処理する毎に、レコーダ２５は、レコーディング・マネージャー２６に対し、ｎｏｔｉｆｙ () 関数を呼び出す。

【００６９】レコーディング・マネージャー２６は、各フレームデータにつき、先にシステムによって登録されたＤＡプラグ・イン・コンポーネント１５を起動する。各ＤＡプラグ・イン・コンポーネント１５は、フレームデータの解析処理を実行し、メタデータを生成する。

【００７０】クエリー＃３に対しては、以下の処理が実行される。

【００７１】ＱＴプラグ・イン・コンポーネント１４は、レコーダ２５に対し、Ｓｔｏｐ () 関数を呼び出し、メディアデータの記録の停止を指示する。

【００７２】レコーダ２５は、メディアデータの記録を停止するとともに、ロガー２４に対し、メディアデータの記録を停止したことを通知する。

【００７３】ＱＴプラグ・イン・コンポーネント１４が、レコーダ２５に対し、ｃｏｍｍｉｔ () 関数を呼び出すことによって、メディアデータの記録が完了する。ここに、ｃｏｍｍｉｔ () 関数に代えて、ａｂｏｒｔ () 関数を呼び出すと、ｓｔａｒｔ () 関数の呼び出し後、このセッション内で記録あるいは生成されたメディアデータおよびメタデータはすべて破棄され、システムは元の状態に復帰する。

【００７４】メタデータ・データベース・パッケージ１１は、メタデータの管理をおこない、メタデータ記憶手段３に記憶されたメタデータ読み出し、メタデータ記憶手段３へのメタデータの記録、メタデータの動的生成およびアクセスにともなうトランザクション処理などの機能を提供するものであり、メタデータ・データベース・パッケージ１１において、メタデータはＪａｖａのクラスとして表わされる。

【００７５】図７は、メタデータの記述クラスのクラスダイアグラムである。

【００７６】図７において、メタデータ・ノード３０は、メタデータを表わすすべてのクラスが共通して、有すべき性質を記述したインターフェイスである。このメタデータ・ノード・インターフェイス３０を継承する形

14

で、メタデータの設計者により、実際のスキーマ定義の実例がＪａｖａのインターフェイスとして定義される。

【００７７】図８は、メタデータの例を示す概念図である。

【００７８】図８に示される例においては、ビデオのシーンの情報を表わすメタデータ、すなわち、ビデオ・シーンが定義されている。ビデオ・シーンは、開始位置、終了位置、代表フレーム番号、シーンを構成するカメラショットなどの属性を有しており、図８に示されるように、それぞれの属性に対するアクセス関数が定義されている。図８において、ビデオ・ショットは、別途定義される他のメタデータを表わしており、この例では、カメラショットに関する情報を表わしている。

【００７９】このようなメタデータを定義するインターフェイスについて、各システムの実装者によって、図７に示される実装クラスの実例３２などのインターフェイスの実装クラスが定義される。この実装クラスにおいて、各関数をどのように実装するかは、各システムの実装者に委ねられており、たとえば、「各属性毎に、変数を用意し、その変数についてアクセスをおこなう」、「ファイルやデータベースにアクセスして、属性値を得る」、「ネットワークでつながれた外部のシステムから属性に関する情報を取得する」、「メディアデータを解析して、動的に属性の値を決定する」などの種々の方法を選択することができる。このように、各システムの実装者が、システム毎に異なる実装をおこなっても、「スキーマ定義の実例」３１によって、共通のインターフェイスを定義しているので、アプリケーションプログラム８をシステム毎に書換える必要はない。

【００８０】図９は、メタデータ・データベース・パッケージ１１の詳細を示すでクラスダイアグラムある。

【００８１】図９において、メタデータ・マネージャー３３は、メタデータ全体についての管理をおこなうための機能を提供するものである。また、トランザクション３４は、メタデータにアクセスする際のトランザクションの管理をおこなうための機能を提供するものである。メタデータ・ノード・ファクトリー３５は、メタデータを定義するインターフェイスについて、各システム固有の実装クラスを選択し、インスタンスを生成する機能を提供するものであり、このクラスを利用することによって、アプリケーションプログラム８あるいはＱＴプラグ・イン・コンポーネント１４は、実装クラスについての知識なしに、適切なクラスのインスタンスを生成することが可能になる。ＤＡプラグ・イン・マネージャー３６は、システムに登録されているＤＡプラグ・イン・コンポーネント１５を管理するものである。また、メタデータ・ノード・ディスクリプター３７およびアトリビュート・ディスクリプター３８は、メタデータの定義に関して、記述するクラスで、「スキーマ定義の実例」３１と同様に、メタデータの設計者によって提供されるもので

15

ある。これらのクラスに記述された情報と、Ｊａｖａのリフレクションａｐｉを用いることによって、個々のメタデータに関する知識なしに、メタデータ全体にわたっての処理を記述することが可能になる。

【００８２】図１０は、メタデータを読み出す手順を示すシーケンス図である。

【００８３】図１０に示されるように、メタデータは、以下のようにして、読み出される。

【００８４】まず、アプリケーションプログラム８から、ＱＴプラグ・イン・コンポーネント１４に問い合わせがおこなわれる。

【００８５】次いで、ＱＴプラグ・イン・コンポーネント１４は、トランザクション３４に対し、ｂｅｇｉｎ（）関数を呼び出し、リードモードで、トランザクションの開始を指示する。

【００８６】ＱＴプラグ・イン・コンポーネント１４は、さらに、メタデータ・マネージャー３３に対し、ｇｅｔＲｏｏｔ（）関数を実行し、すべてのメタデータにアクセスする際の起点となるメタデータであるルートノードを取得する。

【００８７】ＱＴプラグ・イン・コンポーネント１４は、ルートノードから属性を順にたどって行き、必要なメタデータに関する情報を取得する。この際、アクセスしようとしたメタデータが、他のトランザクションにより、ライトモードでアクセスされていた場合には、その間、メタデータに関する情報を取得するための処理は中断され、他のトランザクションが終了した後、処理が再開され、ＱＴプラグ・イン・コンポーネント１４は、必要なメタデータに関する情報を取得する。

【００８８】必要なメタデータに関する情報が取得されると、ＱＴプラグ・イン・コンポーネント１４は、トランザクション３４に対し、ｃｏｍｍｉｔ（）関数を呼び出し、トランザクションを終了させる。

【００８９】図１１は、メタデータを記録する際の手順を示すシーケンス図である。

【００９０】図１１に示されるように、メタデータは、以下のようにして、記録される。

【００９１】まず、アプリケーションプログラム８から、ＱＴプラグ・イン・コンポーネント１４に問い合わせがおこなわれる。

【００９２】次いで、ＱＴプラグ・イン・コンポーネント１４は、メタデータ・ノード・ファクトリー３５に対し、ｃｒｅａｔｅ　Ｎｏｄｅ（）関数を呼び出し、これから記録する新たなメタデータｍ２を生成する。

【００９３】ＱＴプラグ・イン・コンポーネント１４は、さらに、トランザクション３４に対し、ｂｅｇｉｎ（）関数を呼び出し、ライトモードで、トランザクションの開始を指示する。

【００９４】次いで、ＱＴプラグ・イン・コンポーネント１４は、メタデータ・マネージャー３３に対し、ｇｅｔＲｏｏｔ（）関数を実行し、すべてのメタデータにアクセスする際の起点となるメタデータであるルートノードを取得する。この際、アクセスしようとしたメタデータが、他のトランザクションにて、アクセスされていた場合には、その間、ルートノードを取得するための処理は中断され、他のトランザクションが終了した後、処理が再開され、ＱＴプラグ・イン・コンポーネント１４は、ルートノードを取得する。

【００９５】ＱＴプラグ・イン・コンポーネント１４は、ルートノードから属性を順にたどって行き、更新の対象となるメタデータｍ１を取得する。

【００９６】次いで、ＱＴプラグ・イン・コンポーネント１４は、ｍ１で定義されている関数を呼び出して、ｍ２を追加し、あるいは、他の属性値の更新をおこなう。この際、アクセスしようとしたメタデータが、他のトランザクションにて、アクセスされていた場合には、その間、処理は中断され、他のトランザクションが終了した後、処理が再開される。

【００９７】ｍ２を追加し、あるいは、他の属性値の更新をおこなうと、ＱＴプラグ・イン・コンポーネント１４は、トランザクション３４に対し、ｃｏｍｍｉｔ（）関数を呼び出し、トランザクションを終了させる。この際、ｃｏｍｍｉｔ（）関数に代えて、ａｂｏｒｔ（）関数を呼び出すと、このトランザクションで行われた変更はすべて破棄され、トランザクションは開始前の状態に復帰する。

【００９８】図１２は、メタデータに対するアクセスにともなって、動的にメタデータが生成される手順を示すシーケンス図である。

【００９９】図１２に示されるように、メタデータに対するアクセスにともなって、以下のようにして、動的にメタデータが生成される。

【０１００】まず、アプリケーションプログラム８から、ＱＴプラグ・イン・コンポーネント１４に問い合わせがおこなわれる。

【０１０１】次いで、ＱＴプラグ・イン・コンポーネント１４は、トランザクション３４に対し、ｂｅｇｉｎ（）関数を呼び出して、トランザクションの開始を指示する。

【０１０２】ＱＴプラグ・イン・コンポーネント１４は、さらに、メタデータ・マネージャー３３に対し、ｇｅｔＲｏｏｔ（）関数を実行し、すべてのメタデータにアクセスする際の起点となるメタデータであるルートノードを取得する。

【０１０３】ＱＴプラグ・イン・コンポーネント１４は、ルートノードから属性を順にたどって行き、必要なメタデータに関する情報を取得する。

【０１０４】アクセスされた属性が、動的に値が決定されるべきものであった場合には、メタデータ・ノード３０は、ＤＡプラグ・イン・マネージャー３６に問い合わせをおこない、値を求めるためのＤＡプラグ・イン・コンポーネント１５を取得する。

【０１０５】メタデータ・ノード３０は、ＤＡプラグ・イン・コンポーネント１５を呼び出し、アクセスされた属性の値を得る。

【０１０６】ＱＴプラグ・イン・コンポーネント１４は、トランザクション３４に対し、ｃｏｍｍｉｔ（）関数を呼び出し、トランザクションを終了させる。

【０１０７】図１３は、クウェリー・インターフェイス・パッケージ１２のクラスダイアグラムである。

【０１０８】クウェリー・インターフェイス・パッケージ１２は、メディアデータ・データベース・パッケージ１０およびメタデータ・データベース・パッケージ１１の機能を利用して、メタデータおよび／またはメディアデータを利用したサービスをアプリケーションプログラム８に対して提供するものである。アプリケーションプログラム８に対して提供されるサービスとは、「メディアデータ視聴機器に記録されているコンテンツの一覧を表示し、ユーザーに選択させる」、「コンテンツの部分に対して、注釈を、メタデータとして、付加し、および／または、表示する」「コンテンツの視聴をおこないながら、注釈が付加された部分にさしかかると、画面の隅にマークを表示し、あるいは、音声を鳴らすなどして、ユーザーに注釈の存在を知らせる」、「注釈が付加された部分をピックアップして、コンテンツのダイジェスト版を作成し、再生する」など、複数のアプリケーションプログラム８が共通して利用できるハイレベルな機能を切り出し、簡単なａｐｉによって、使用できるようにしたものである。

【０１０９】本実施態様においては、これらのサービスを、ＱＴプラグ・イン・コンポーネント１４として、パッケージ本体から分離し、クウェリー・インターフェイス・パッケージ１２自体は、ＱＴプラグ・イン・コンポーネント１４の管理および／またはルックアップ機能のみを提供するように構成されている。これによって、後に、種々のサービスを動的に追加することが可能になる。

【０１１０】本実施態様において、ＱＴプラグ・イン・コンポーネント１４は、Ｊａｖａにおいての標準的なコンポーネントモデルであるＪａｖａＢｅａｎｓとして作成される。ＱＴプラグ・イン・コンポーネント１４自身をＧＵＩの部品として作ることもでき、これにより、ＱＴプラグ・イン・コンポーネントの一例を示す図１４に示されるように、複数のＱＴプラグ・イン・コンポーネントを組み合わせるだけで、アプリケーションプログラム８に必要とされる機能をＧＵＩ込みで、簡単に構築することが可能となる。

【０１１１】図１４において、ＱＴプラグ・イン・マネージャー４０は、ＱＴプラグ・イン・コンポーネント１４の管理をおこない、アプリケーションプログラム８の要求にしたがって、適切なＱＴプラグ・イン・コンポーネント１４を探し出し、インスタンシエートする機能を提供するものである。

【０１１２】また、ＱＴプラグ・イン・ディスクリプター４１は、各ＱＴプラグ・イン・コンポーネント１４についての情報を取得する方法を記述したインターフェイスで、ＱＴプラグ・イン・コンポーネント１４の開発者によって実装され、提供される。図１５は、ＱＴプラグ・イン・ディスクリプター４１の一例を示すものであり、図１５に示されるＱＴプラグ・イン・ディスクリプター４１は、ＱＴプラグ・イン・コンポーネントが提供するＪａｖａのクラス／インターフェイスとして指定されるサービスの種類、開発者、バージョン情報などの種々の情報を含んでおり、これらの情報に基づいて、アプリケーションプログラム８は必要なＱＴプラグ・イン・コンポーネントを検索するように構成されている。

【０１１３】図１４において、ＱＴプラグ・イン・テンプレート４２は、アプリケーションプログラム８がＱＴプラグ・イン・コンポーネントを検索する際に、必要とするＱＴプラグ・イン・コンポーネントを指定するのに用いられるものである。ＱＴプラグ・イン・コンポーネントを検索するときには、このＱＴプラグ・イン・テンプレート４２と前述したＱＴプラグ・イン・ディスクリプター４１の値のマッチングがおこなわれる。その際、ｎｕｌｌ値はワイルドカードとして扱われる。

【０１１４】図１６は、クウェリー・インターフェイス・パッケージ１２を使用する際の手順を示すシーケンス図である。

【０１１５】図１６に示されるように、アプリケーションプログラム８の実行に先立ち、システムは利用可能なＱＴプラグ・イン・コンポーネント１４について、対応するＱＴプラグ・イン・ディスクリプター４１の実装クラスを、ＱＴプラグ・イン・マネージャー４０に登録する。

【０１１６】続いて、アプリケーションプログラム８を実行し、まず、アプリケーションプログラム８において、必要とするＱＴプラグ・イン・コンポーネント１４の情報を記述するＱＴプラグ・イン・テンプレート４２を作成する。

【０１１７】次いで、作成したＱＴプラグ・イン・テンプレート４２を引数として、ＱＴプラグ・イン・マネージャー４０に対し、ｌｏｏｋｕｐ（）関数を呼び出す。

【０１１８】ＱＴプラグ・イン・マネージャー４０では、与えられたＱＴプラグ・イン・テンプレート４２と、その時点で、登録されているすべてのＱＴプラグ・イン・ディスクリプター４１とのマッチングがおこなわれ、マッチしたものについて、そのＱＴプラグ・イン・コンポーネント１４のインスタンスを作成し、アプリケーションプログラム８に返す。

【０１１９】アプリケーションプログラム８は、自らの

17

18

19

目的に合致するように、ＱＴプラグ・イン・コンポーネント１４のプロパティを設定し、あるいは、イベントリスナーの登録をおこなう。

【０１２０】アプリケーションプログラム８は、ＱＴプラグ・イン・コンポーネント１４に対して、問い合わせをおこない、必要なサービスを受ける。これにともなって、前述のように、メディアデータ・データベース・パッケージ１０およびメタデータ・データベース・パッケージ１１に対するアクセスがおこなわれる。

【０１２１】本実施態様によれば、メタデータ管理手段４によって管理されるメタデータを用いることによって、メディアデータに対して、ある俳優の映っている部分などの意味的なアクセスをおこなうことが可能になるとともに、メディアデータ管理手段２の機能によって、メディアデータのダイジェストの生成などのメディアデータの合成処理を実行することが可能になり、かつ、問い合わせ手段６の機能によって、これらの機能を有したアプリケーションプログラムを効率的に開発することができる。

【０１２２】また、本実施態様によれば、メディアデータ管理手段２において、メディアデータに対するアクセスの履歴を記録する際に、アクセス履歴情報をメタデータに変換し、メタデータ管理手段４に記録するように構成されているため、ユーザーの視聴履歴情報についても、他のメタデータと同様に、ミドルウェアおよびアプリケーションプログラムから統一的に扱うことが可能となり、アプリケーションプログラムのサイズを小さく保つことができ、また、アクセス履歴情報をメタデータに変換することによって、同じミドルウェアを搭載している他の視聴機器の間での視聴履歴情報の交換が可能になる。

【０１２３】さらに、本実施態様によれば、メディアデータ管理手段２において、メディアデータの記録をおこなう際に、入力されたメディアデータについて、あらかじめ登録されたデータ解析処理を適用することによって、メタデータを作成し、メタデータ管理手段４に記録するように構成されているから、メディアデータとともに、メタデータが提供されない場合でも、メディアデータ視聴機器側で、メタデータを生成することが可能となり、メタデータを利用したアプリケーションプログラムを動作させることが可能になる。

【０１２４】また、本実施態様によれば、メタデータ管理手段４が有するメタデータを動的に生成する機能を、独立したＤＡプラグ・イン・コンポーネント１６として分離し、ＤＡプラグ・イン・コンポーネント１６をシステムに動的に組み込む機能、使用しなくなったＤＡプラグ・イン・コンポーネント１６を破棄する機能およびすでに組み込まれているＤＡプラグ・イン・コンポーネント１６のうち、アプリケーションプログラムの目的に合致したＤＡプラグ・イン・コンポーネント１６を自動的

20

に選択し、実行する機能を提供するように構成されているから、メタデータを動的に組み込むことが可能になり、種々のソースから得られるメタデータをミドルウェアが解釈できる形に変換して、統一的に扱うことが可能となるとともに、将来、新しいアルゴリズム／フォーマットが開発された場合でも、ミドルウェア／システムを変更することなしに、それらの新しいアルゴリズム／フォーマットを導入することができる。

【０１２５】さらに、本実施態様によれば、問い合わせ手段６が有するメディアデータ管理手段２およびメタデータ管理手段４へのアクセス処理を代行し、アプリケーションプログラム８に対して、より抽象度の高いアクセスインターフェイスを提供する機能を独立したＱＴプラグ・イン・コンポーネント１４として分離し、ＱＴプラグ・イン・コンポーネント１４をシステムに動的に組み込む機能、使用しなくなったＱＴプラグ・イン・コンポーネント１４を破棄する機能およびすでに組み込まれているプラグ・イン・コンポーネントのうち、アプリケーションプログラムから提示された条件に合致したプラグ・イン・コンポーネントを検索する機能を提供するように構成されているから、ミドルウェアの機能を必要に応じて拡張することが可能となり、アプリケーションプログラム８の多彩な要求に応えることができ、また、アプリケーションプログラム８の開発を、メタデータ管理手段４やメディアデータ管理手段２の機能を用いて、基本的な機能を開発する開発者と、それらの機能を利用して、アプリケーションプログラム８全体の機能を開発する開発者とで分業することが可能となって、アプリケーションプログラム８の開発効率をさらに向上させることができる。

【０１２６】本発明は、以上の実施態様に限定されることなく、特許請求の範囲に記載された発明の範囲内で種々の変更が可能であり、それらも本発明の範囲内に包含されるものであることはいうまでもない。

【０１２７】

【発明の効果】本発明によれば、メタデータを用いることによって、メディアデータに対して、ある俳優の映っている部分などの意味的なアクセスをおこなうことを可能とするとともに、メディアデータのダイジェストの生成などのメディアデータの合成処理を実行することができ、かつ、これらの機能を有したアプリケーションプログラムを効率的に開発することができるミドルウェアおよびそれを用いたメディアデータ視聴機器を提供することが可能となる。

【図面の簡単な説明】

【図１】図１は、本発明の実施態様にかかるミドルウエアを含むメディアデータ視聴機のブロックダイアグラムである。

【図２】図２は、システムを構成するソフトウエアの階層図である。

21

【図３】図３、本発明の好ましい実施態様にかかるミドルウエアのパッケージ構成を示すパッケージ図である。

【図４】図４は、メディアデータ・データベース・パッケージの詳細を示すクラスダイアグラムである。

【図５】図５は、メディアデータを再生する手順を示すシーケンス図である。

【図６】図６は、メディアデータを記録する際の手順を示すシーケンス図である。

【図７】図７は、メタデータの記述クラスのクラスダイアグラムである。

【図８】図８は、メタデータの例を示す概念図である。

【図９】図９は、メタデータ・データベース・パッケージの詳細を示すクラスダイアグラムである。

【図１０】図１０は、メタデータを読み出す手順を示すシーケンス図である。

【図１１】図１１は、メタデータを記録する際の手順を示すシーケンス図である。

【図１２】図１２は、メタデータに対するアクセスにともなって、動的にメタデータが生成される手順を示すシーケンス図である。

【図１３】図１３は、クウェリー・インターフェイス・パッケージのクラスダイアグラムである。

【図１４】図１４は、ＱＴプラグ・イン・コンポーネントの一例を示す概念図である。

【図１５】図１５は、ＱＴプラグ・イン・ディスクリプター・インターフェイス４１の記述例である。

【図１６】図１６は、クウェリー・インターフェイス・パッケージ１２を使用する際の手順を示すシーケンス図である。

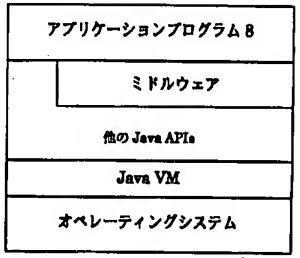
【符号の説明】

１　メディアデータ記憶手段
２　メディアデータ管理手段

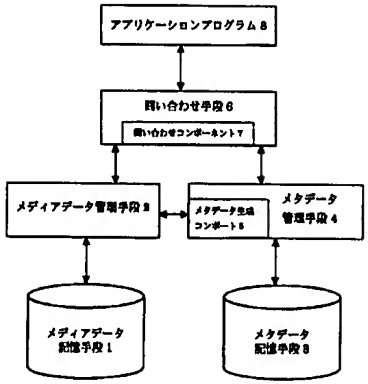
22

＊３　メタデータ記憶手段
４　メタデータ管理手段
５　メタデータ生成コンポーネント
６　問い合わせ手段
７　問い合わせコンポーネント
８　アプリケーションプログラム
１０　メディアデータ・データベース・パッケージ
１１　メタデータ・データベース・パッケージ
１２　クウェリー・インターフェイス・パッケージ
１３　ユーティリテイ・パッケージ
１４　ＱＴプラグ・イン・コンポーネント
１５　ＤＡプラグ・イン・コンポーネント
２０　Ｊａｖａｘ．ｍｅｄｉａ．パッケージ
２１　Ｐｒｏｃｅｓｓｏｒ　実装クラス
２２　プロセッサ
２３　プレイリスト・エントリー・クラス
２４　ロガー
２５　レコーダ
２６　レコーディング・マネージャー
２７　メディア・マネージャー
３０　メタデータ・ノード
３１　スキーマ定義の実例
３２　実装クラスの実例
３３　メタデータ・マネージャー
３４　トランザクション
３５　メタデータ・ノード・ファクトリー
３６　ＤＡプラグ・イン・マネージャー
３７　メタデータ・ノード・ディスクリプター
３８　アトリビュート・ディスクリプター
４０　ＱＴプラグ・イン・マネージャー
４１　ＱＴプラグ・イン・ディスクリプター
＊４２　ＱＴプラグ・イン・テンプレート

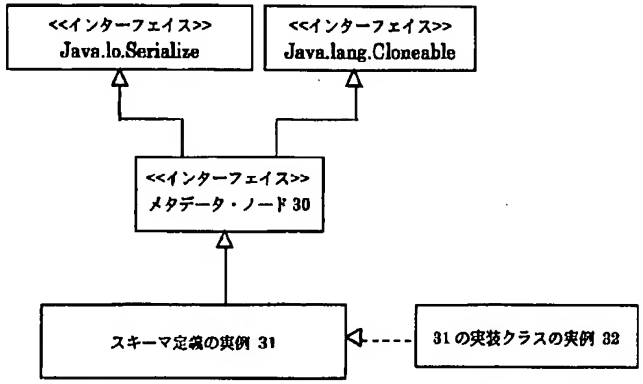
【図２】



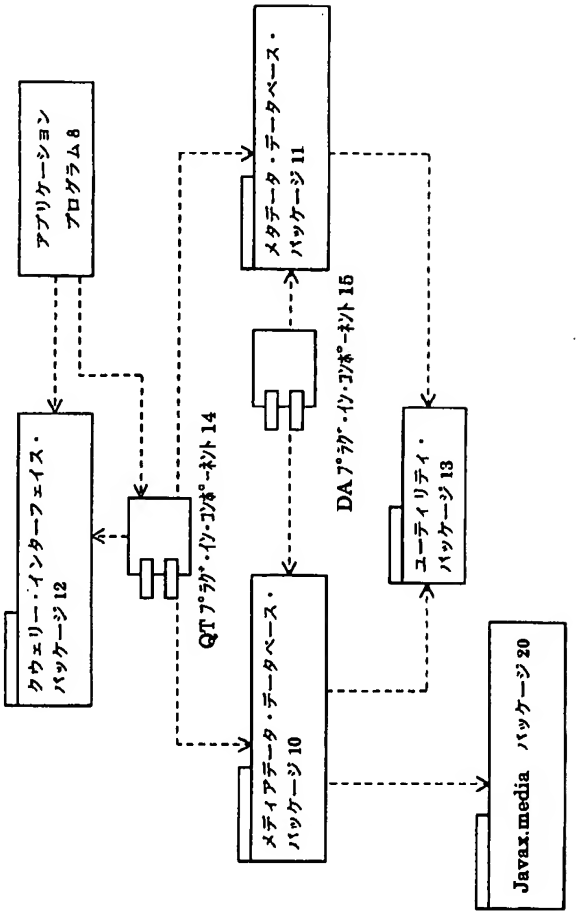
【図1】


アプリケーションプログラム 8
問い合わせ手段 6
問い合わせコンポーネント 7
メディアデータ管理手段 3
メタデータ生成コンポーネント 5
メタデータ管理手段 4
メディアデータ記憶手段 1
メタデータ記憶手段 2

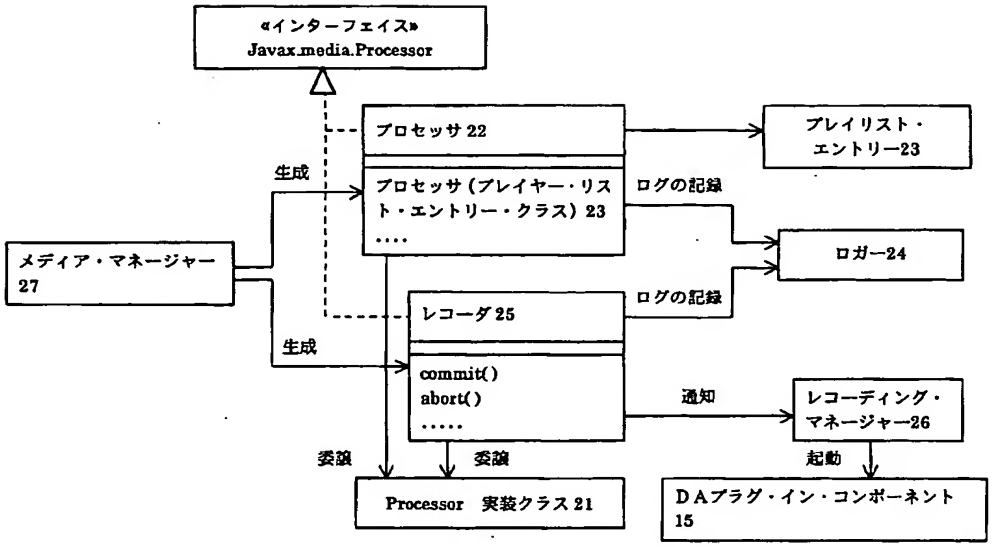
【図7】


<<インターフェイス>>
Java.lo.Serialize
<<インターフェイス>>
Java.lang.Cloneable
<<インターフェイス>>
メタデータ・ノード 30
スキーマ定義の実例 31
31 の実装クラスの実例 32

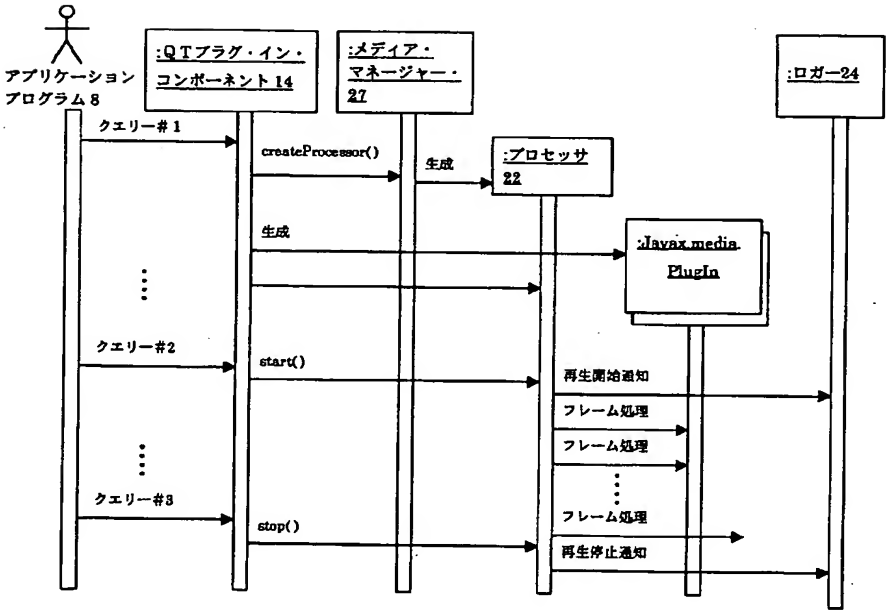
【図3】


アプリケーション
プログラム 8
メタデータ・データベース・
パッケージ 11
DA プラガブル・インタフェース 15
クウェリー・インターフェイス・
パッケージ 12
QT プラガブル・インタフェース 14
ユーティリティ・
パッケージ 13
メディアデータ・データベース・
パッケージ 10
Javax.media パッケージ 20

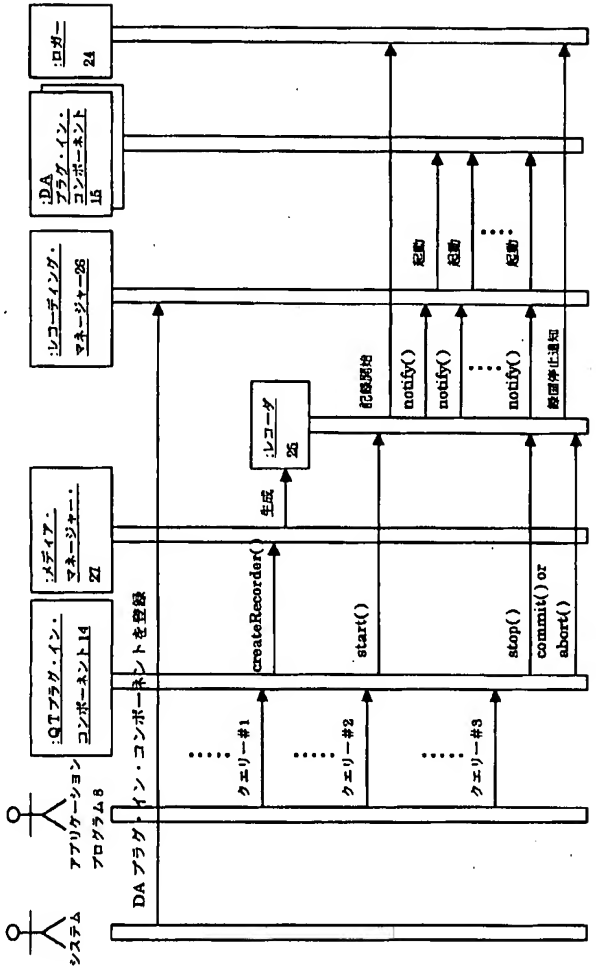
[図4]


«インターフェイス»
Javax.media.Processor
プロセッサ 22
プレイリスト・
エントリー23
生成
プロセッサ（プレイヤー・リス
ト・エントリー・クラス）23
....
ログの記録
メディア・マネージャー
27
ロガー24
ログの記録
レコーダ 25
生成
commit()
abort()
.....
通知
レコーディング・
マネージャー26
委譲
委譲
起動
Processor 実装クラス 21
DAプラグ・イン・コンポーネント
15

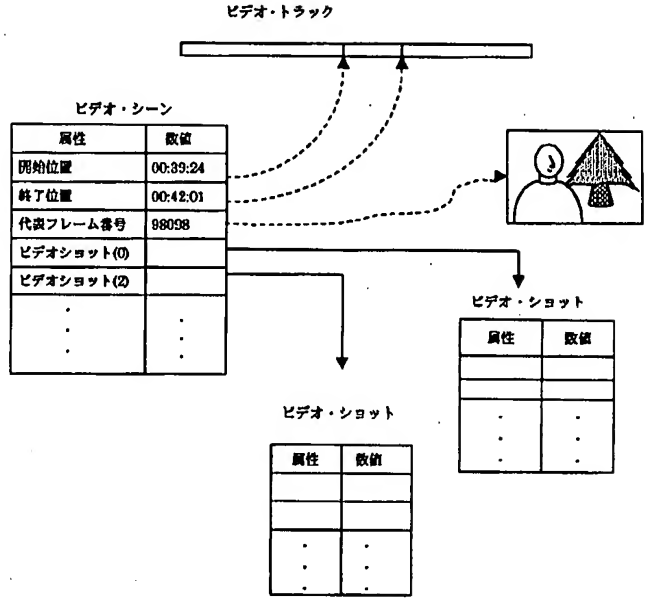
[図5]


アプリケーション
プログラム 8
:QTプラグ・イン・
コンポーネント 14
:メディア・
マネージャー・
27
:ロガー24
クエリー# 1
createProcessor()
生成
:プロセッサ
22
生成
:Javax.media.
PlugIn
クエリー#2
start()
再生開始通知
フレーム処理
フレーム処理
クエリー#3
stop()
フレーム処理
再生停止通知

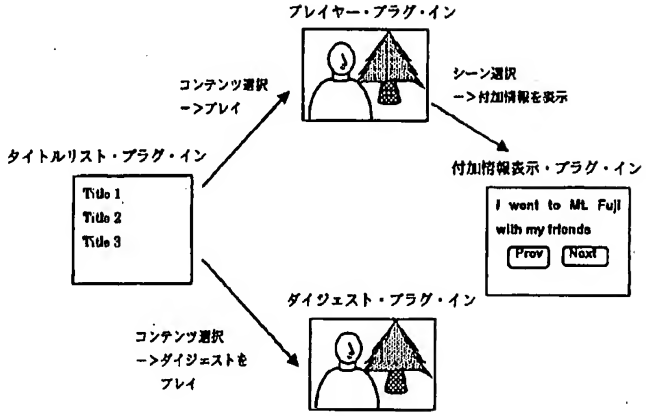
【図６】


:ロガー 24
:DA プラグ・イン・コンポーネント 15
:レコーディング・マネージャー 26
:レコーダ 25
:メディア・マネージャー 27
:QTプラグ・イン・コンポーネント 14
アプリケーション プログラム 8
システム
DA プラグ・イン・コンポーネントを登録
生成
createRecorder()
start()
stop()
commit() or abort()
記録開始
記録停止通知
notify()
起動
クエリー#1
クエリー#2
クエリー#3

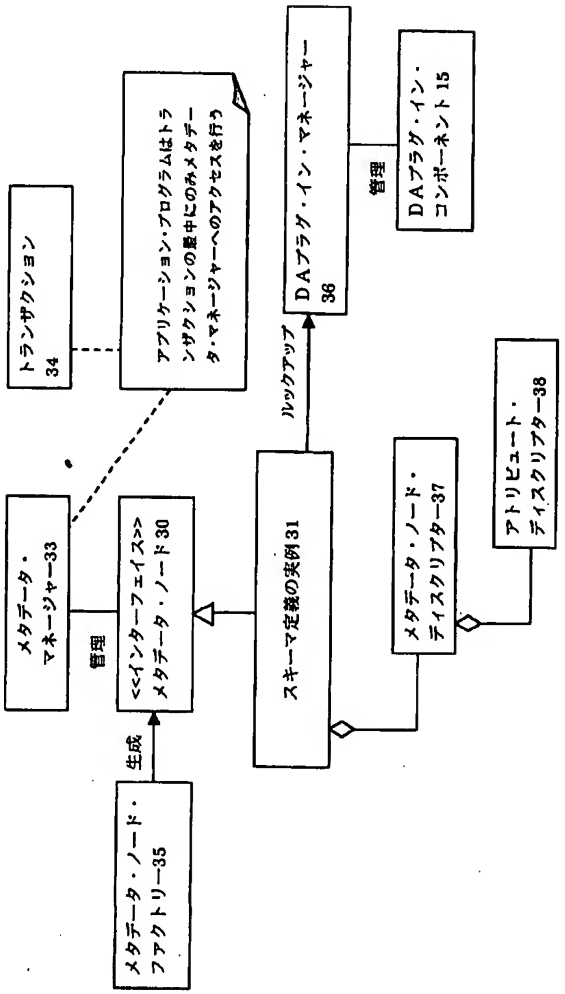
【図８】


ビデオ・トラック
ビデオ・シーン
属性 数値
開始位置 00:39:24
終了位置 00:42:01
代表フレーム番号 98098
ビデオショット(0)
ビデオショット(2)
ビデオ・ショット
属性 数値
ビデオ・ショット
属性 数値

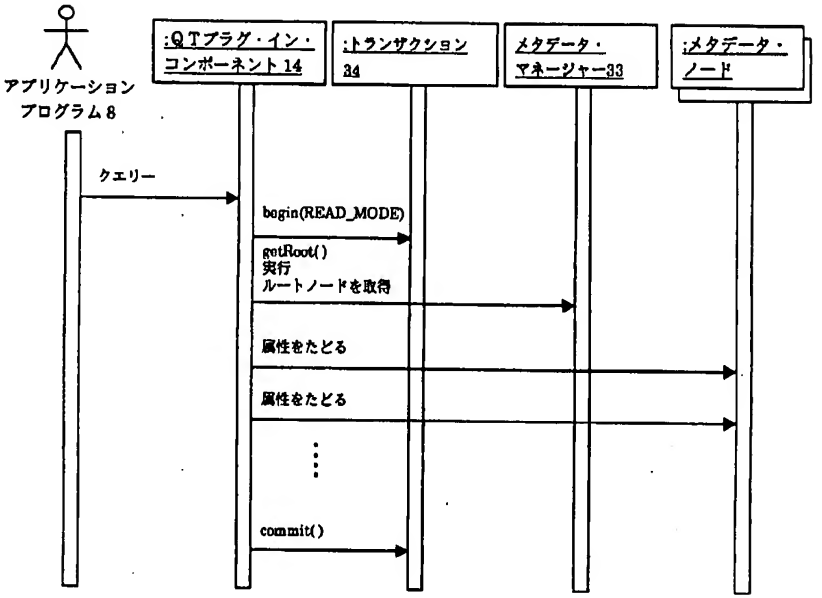
【図１４】


プレイヤー・プラグ・イン
コンテンツ選択
->プレイ
シーン選択
->付加情報を表示
タイトルリスト・プラグ・イン
Title 1
Title 2
Title 3
付加情報表示・プラグ・イン
I went to Mt. Fuji
with my friends
Prev
Next
ダイジェスト・プラグ・イン
コンテンツ選択
->ダイジェストを
プレイ

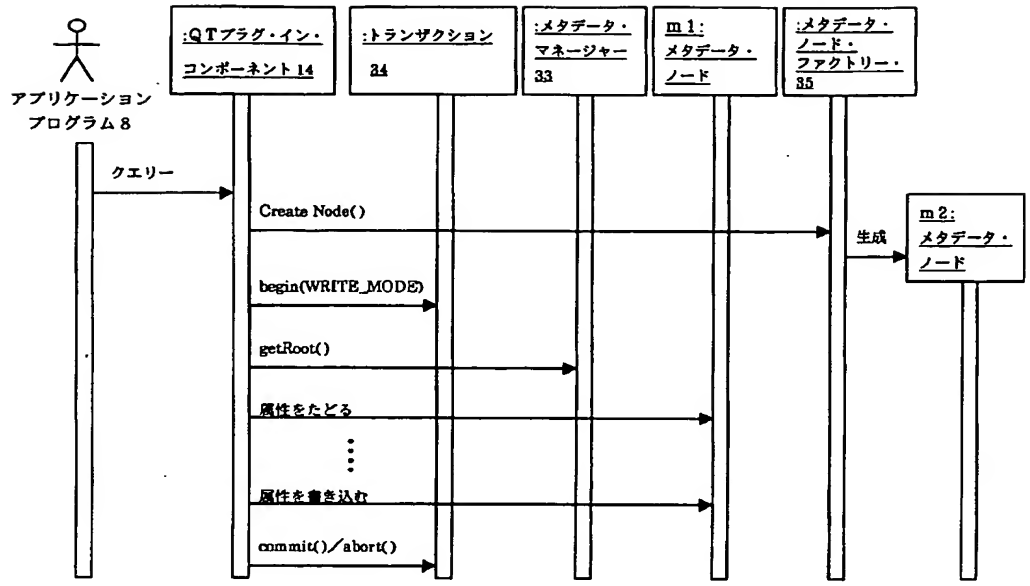
【図９】


トランザクション
34
アプリケーション・プログラムはトランザクションの最中にのみメタデータ・マネージャーへのアクセスを行う
DAプラグ・イン・マネージャー
36
管理
DAプラグ・イン・コンポーネント15
ルックアップ
メタデータ・マネージャー33
管理
<<インターフェイス>>
メタデータ・ノード30
スキーマ定義の実例31
メタデータ・ノード・ディスクリプター37
アトリビュート・ディスクリプター38
生成
メタデータ・ノード・ファクトリー35

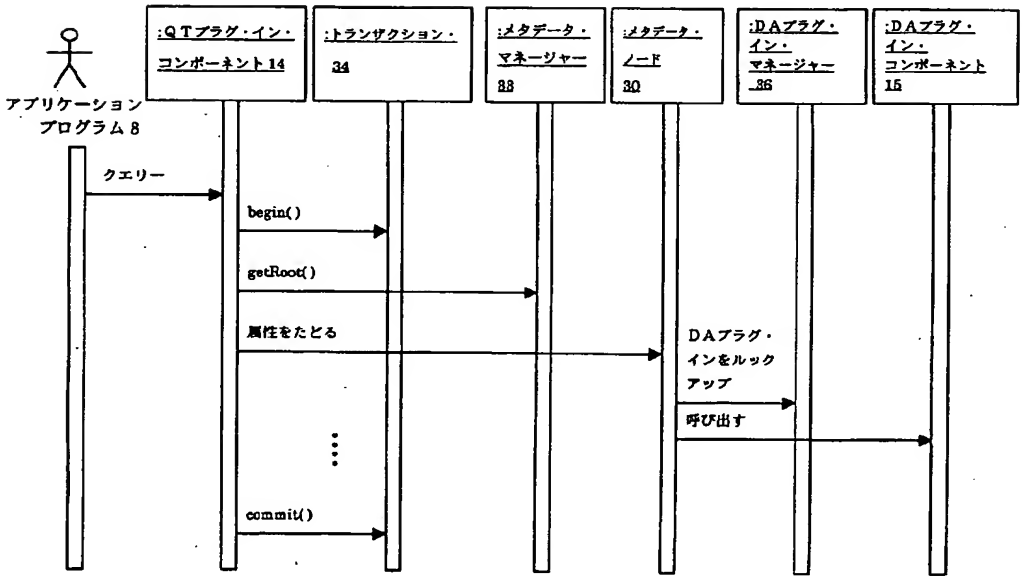
【図１０】


アプリケーション
プログラム8
:QTプラグ・イン・コンポーネント14
:トランザクション34
メタデータ・マネージャー33
:メタデータ・ノード
クエリー
begin(READ_MODE)
getRoot()
実行
ルートノードを取得
属性をたどる
属性をたどる
commit()

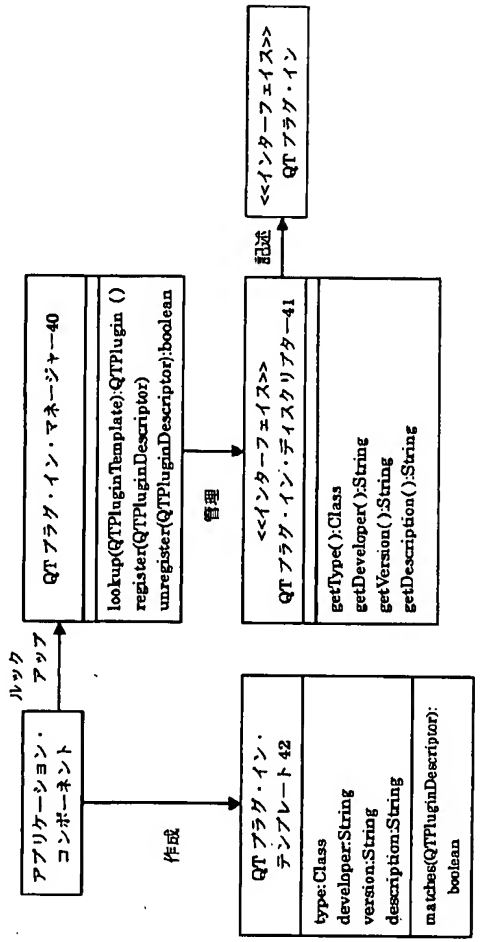
[図11]


アプリケーション
プログラム8
:QTプラグ・イン・
コンポーネント14
:トランザクション
34
:メタデータ・
マネージャー
33
m1:
メタデータ・
ノード
:メタデータ・
ノード・
ファクトリー・
35
クエリー
Create Node()
生成
m2:
メタデータ・
ノード
begin(WRITE_MODE)
getRoot()
属性をたどる
属性を書き込む
commit()／abort()

[図12]


アプリケーション
プログラム8
:QTプラグ・イン・
コンポーネント14
:トランザクション・
34
:メタデータ・
マネージャー
38
:メタデータ・
ノード
30
:DAプラグ・
イン・
マネージャー
36
:DAプラグ・
イン・
コンポーネント
15
クエリー
begin()
getRoot()
属性をたどる
DAプラグ・
インをルック
アップ
呼び出す
commit()

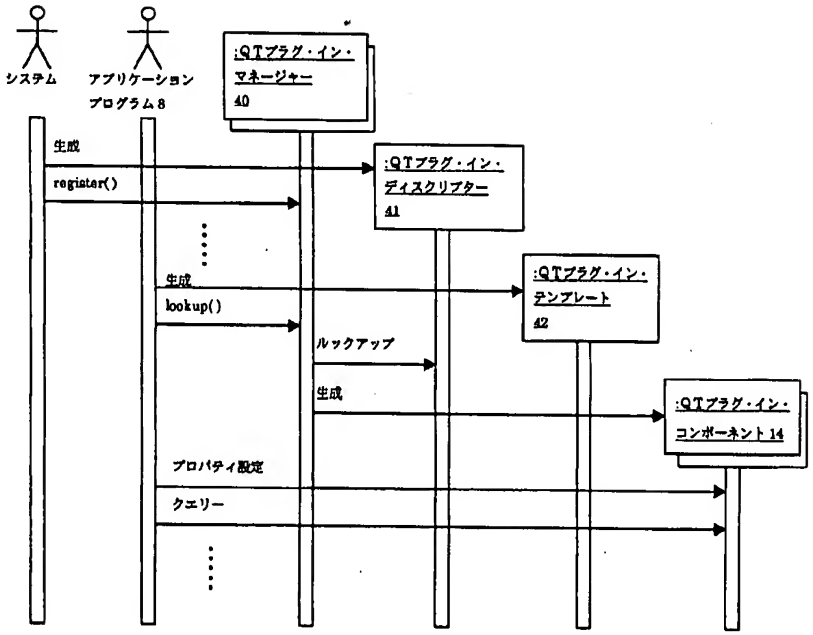
【図13】



【図15】

```
import queryInterface.QTPlugInDescriptor;

public class ConcreteQTPlugInDescriptor
    implements QTPlugInDescriptor {

  public Class getType() {
    return ConcreteQTPlugIn.class;
  }

  public String getDeveloper() {
    return "XXX Corporation";
  }

  public String getVersion() {
    return "2.3_p14";
  }

  public String getAuthKey() {
    return "9598468074f2691f0630cce56dc45502";
  }

  public String getDescription() {
    return "Concrete QTPlugIn component ";
           + "designed as sample.";
  }
}
```

【図１６】


システム
アプリケーション
プログラム８
:ＱＴプラグ・イン・
マネージャー
40
:ＱＴプラグ・イン・
ディスクリプター
41
:ＱＴプラグ・イン・
テンプレート
42
:ＱＴプラグ・イン・
コンポーネント 14
生成
register()
生成
lookup()
ルックアップ
生成
プロパティ設定
クエリー